Communication avoiding low rank approximation based on QR with tournament pivoting

Matthias Beaupère, Laura Grigori

Communication avoiding low rank approximation based on QR with tournament pivoting

Matthias Beaupere* Laura Grigori†

Abstract

We introduce a parallel algorithm for computing the low rank approximation A_k of a large matrix A which minimizes the number of messages exchanged between processors (modulo polylogarithmic factors) and has guarantees for the approximations of the singular values of A provided by A_k. This operation is essential in many applications in scientific computing and data analysis when dealing with large data sets. Our algorithm is based on QR factorization that consists in selecting a subset of columns from the matrix A that allow to approximate the range of A, and then projecting the columns of A on a basis of the subspace spanned by those columns. The selection of columns is performed by using tournament pivoting, a strategy introduced previously for matrices partitioned into blocks of columns. This strategy is extended here to matrices partitioned along both dimensions that are distributed on a two-dimensional grid of processors, and also to tournaments with more general reduction trees. Performance results show that the algorithm scales well on up to 1024 cores of 16 nodes.

1 Introduction

In this paper we focus on computing the low rank approximation of a large matrix in a parallel environment. This operation appears in a variety of domains in scientific computing and data analysis. The problem to solve is the following. Given a matrix $A \in \mathbb{R}^{m\times n}$, we are looking for a rank k matrix A_k that approximates well A. The truncated singular value decomposition (SVD) $A_{opt,k}$ provides the best rank-k approximation in terms of Frobenius and L2 norm [9], e.g. $\|A - A_{opt,k}\|_2 = \sigma_{k+1}(A)$ where σ_{k+1} is the $k+1$ singular value of A. However, it is expensive to compute and several algorithms have been introduced in the literature to approximate the SVD.

The idea underlying many algorithms for computing a low rank approximation is to construct a low dimensional subspace $X = range(A\Pi_1)$ that approximates well the range of A, where $\Pi_1 \in \mathbb{R}^{n\times k}$. Given $Q_1 \in \mathbb{R}^{m\times k}$ an orthogonal basis of $range(A\Pi_1)$, the low rank approximation is computed as the projection of the columns of A on this basis, $A_k = Q_1 Q_1^T A$. We consider in this paper the case in which the low dimensional subspace is obtained by selecting k columns of A and the low rank approximation is obtained through the QR factorization with column pivoting. This factorization computes the decomposition of $A\Pi$ as,

$$A\Pi = QR = \begin{bmatrix} Q_1 & Q_2 \end{bmatrix} \begin{bmatrix} R_{11} & R_{12} \\ & R_{22} \end{bmatrix}, \tag{1}$$

where $\Pi \in \mathbb{R}^{n\times n}$, $\Pi = (\Pi_1, \Pi_2)$ is a permutation matrix, $Q \in \mathbb{R}^{m\times m}$ is orthogonal, $R \in \mathbb{R}^{m\times n}$, $Q_1 \in \mathbb{R}^{m\times k}$, and $R_{11} \in \mathbb{R}^{k\times k}$ is upper triangular. A low rank approximation is obtained as,

$$A_k = Q_1 Q_1^T A = Q_1 \begin{bmatrix} R_{11} & R_{12} \end{bmatrix}. \tag{2}$$

There are several approaches for choosing a permutation Π that reduces the error of the low rank approximation $\|A - A_k\|_2$. One approach, referred to as QRCP, is to select the column with the largest norm [1, 11] at each step of the QR factorization. Another approach is to reject $n-k$ columns one-by-one using the lowest singular value of the non-rejected columns [3, 4]. This strategy is sometimes referred to as reverse

*ALPINES, Inria Paris, France (matthias.beaupere@inria.fr)
†ALPINES, Inria Paris, France (laura.grigori@inria.fr)

pivoting [13, 12, 15]. Strong rank revealing QR (strong RRQR) [11] is another approach that relies on computing QRCP followed by a number of additional column permutations. For a given $k \leq \min(m, n)$, it is shown in [11] that for any $1 \leq j \leq n-k$, $1 \leq i \leq k$, and a constant $f > 1$,

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})} \leq \sqrt{1 + kf^2(n-k)}, \quad 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq \sqrt{1 + kf^2(n-k)}, \tag{3}$$

Following the terminology in [6], a low rank approximation that satisfies the bounds above is called spectrum preserving (equation (3) left) and kernel approximation (equation (3) right) of A. We use here a slightly modified definition of the spectrum preserving property, which bounds the ratio $\sigma_i(A)/\sigma_i(R_{11})$ instead of $\sigma_i(A)/\sigma_i(A_k)$ in [6]. We note that $\sigma_j(R_{22}) = \sigma_j(A - A_k)$. The lower bounds of 1 are satisfied for any permutation Π through the interlacing property of singular values.

While QRCP and strong RRQR provide in practice accurate approximations of the SVD, they are expensive in terms of communication when computed in a parallel environment. If the matrix A is distributed over P processors, QRCP requires exchanging at least $k \log_2 P$ messages for computing a rank-k approximation. The additional permutations performed in strong RRQR lead to an even larger number of messages. Communication-avoiding RRQR (CARRQR) introduced in [7] (and presented later in this paper in algorithm 1) allows selecting k columns from A using only $\log_2 P$ messages through the usage of tournament pivoting. It is thus communication optimal in terms of number of messages and can be more efficient than QRCP and strong RRQR on a parallel computer. Indeed, communication is the major bottleneck in achieving performance on massively parallel computers. Tournament pivoting considers that the matrix A is partitioned into blocks of columns and performs the selection of k columns as a reduction with strong RRQR being the operator applied at each step of the reduction. We refer to this strategy as 1Dc-TP.

In this paper we introduce QRTP, an algorithm that allows to compute in parallel the rank-k approximation of a matrix A partitioned into $P_r \times P_c$ blocks,

$$A = \begin{bmatrix} A_{11} & \dots & A_{1P_c} \\ \vdots & \ddots & \vdots \\ A_{P_r1} & \dots & A_{P_rP_c} \end{bmatrix}. \tag{4}$$

This algorithm relies on extending tournament pivoting to allow the selection of k columns from a matrix partitioned as in equation (4). We refer to this pivoting strategy as 2D tournament pivoting (2D TP for short). We first discuss tournament pivoting for a matrix partitioned into P_r blocks of rows, as for example the first block column of A, $[A_{11}; \dots; A_{P_r1}]$, which allows selecting k columns from the selection of k sub-columns in each block A_{i1} through strong RRQR. We refer to this selection as 1Dr-TP. Then by defining a reduction tree and by combining 1Dr-TP with 1Dc-TP, k columns of A can be selected from the initial selection of k sub-columns in each block A_{ij}. We show that the obtained factorization is spectrum preserving and kernel approximation, satisfying with the notation from (1) the bounds (see for more details section 4.1),

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})} \leq \sqrt{1 + kf_{TP}^2(nP_c - k)}, \quad 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{j+k}(A)} \leq \sqrt{1 + kf_{TP}^2(nP_c - k)}, \tag{5}$$

where

$$f_{TP} = \sqrt{P} P_r k^{\log_2(P)} f^{\log_2(P)+1}.$$

This algorithm can be efficiently executed in parallel on P processors, where $P = P_r \cdot P_c$, and requires exchanging $\log_2 P(1 + \log_2 P_r)$ messages.

The paper is organized as follows. Section 2 gives background and past work leading to 1Dc-TP, which generalizes the bounds of tournament pivoting in [7] to more general reduction trees. Section 3 introduces 1Dr-TP, an analogous strategy that allows to select k columns from a matrix partitioned into blocks of rows. Section 4 introduces QRTP (2D tournament pivoting), the main algorithm of this work, which relies on a combination of 1Dc-TP and 1Dr-TP. We show that QRTP provides a spectrum revealing and kernel approximation of the original matrix and works well in practice. Section 6 presents the parallel design of QRTP as well as its estimated parallel cost. Parallel performance results from Section 7 show that for a 100 000 $\times$ 100 000 matrix, QRTP leads to a speedup of 6 when increasing the number of cores from 128 to 1024 (a factor of 8).

2 Notation and tournament pivoting for block column partitioned matrices

This section introduces some notation and presents past work on the problem of low rank approximation using QR decomposition.

2.1 Notation

Throughout the paper we use the following notations.

- $\mathbb{R}^{m\times n}$ is the set of matrices having m rows and n columns.
- For any matrix A, $A[i:j,k:l]$ denotes the submatrix containing rows i to j and columns k to l of A, $A[:,k:l]$ denotes the submatrix containing columns k to l of A, $A[:,:l]$ denotes the submatrix containing the l first columns of A, and $A[:,j]$ denotes the submatrix containing only the column of A indexed by j.
- For any matrix A, and a set of indices I, $A[:,I]$ denotes the submatrix containing columns of indices I of A.
- For any matrix $A \in \mathbb{R}^{m\times n}$, we define $\omega_i(A) = \frac{1}{\|A^{-1}[i,:]\|}$ and $\lambda_i(A) = \|A[:,j]\|$
- $\mathcal{O}$ denotes a matrix containing only zeroes.
- $\mathcal{I}$ denotes the identity matrix.
- For any matrices A_1 and A_2 having the same number of rows, $[A_1;A_2]$ denotes the vertical concatenation of A_1 and A_2.
- For any matrix A, $\sigma_k(A)$ denotes the kth largest singular value of A

2.2 Background on SVD and strong RRQR

The truncated SVD is defined as $A_{opt,k} = U_k\Sigma_k V_k^T$, where $\Sigma_k \in \mathbb{R}^{k\times k}$ is a diagonal matrix formed by the k leading singular values of A, $\sigma_1(A),\ldots,\sigma_k(A)$, $U_k \in \mathbb{R}^{m\times k}$ and $V_k \in \mathbb{R}^{n\times k}$ are the corresponding left and right singular vectors, respectively. The truncated SVD provides the best low rank approximation in terms of Frobenius norm and L2 norm [9], with $\|A - A_{opt,k}\|_2 = \sigma_{k+1}(A)$ and $\|A - A_{opt,k}\|_F = \sqrt{\sum_{i=k+1}^{\min(m,n)} \sigma_i^2(A)}$.

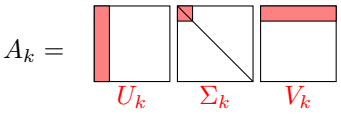

Figure 1: Truncated singular value decomposition

As described in Section 1, the QR decomposition with column pivoting relies on a permutation matrix Π and computes the decomposition $A\Pi = QR$. There are several approaches for choosing a permutation Π. In our implementation we use `DLAQPS` from LAPACK [14, 8] which implements a block QRCP based on the original algorithm from Businger and Golub [1]. However, our theoretical results are based on the bounds obtained by strong RRQR, that we present in the following two theorems.

Theorem 1. (Lemma 3.1 and Algorithm 4 in [11]) *Let A be an $m\times n$ matrix and $1 \le k \le \min(m,n)$. For any $f > 1$ there exists a permutation matrix Π such that the decomposition described in (1) verifies for all $(i,j) \in [1,k]\times[1,n-k]$,*

$$(R_{11}^{-1}R_{12})_{ij}^2 + \omega_i^2(R_{11})\gamma_j^2(R_{22}) \le f^2. \tag{6}$$

A factorization satisfying (6) is thus called a strong RRQR factorization. We also use the relaxed form by summing over i (Corollary 2.3 in [7]),

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq kf^2, \tag{7}$$

leading to the following theorem.

Theorem 2. (Theorem 3.2 in [11]) and (Theorem 2.4 in [7]) *Let A be an $m \times n$ matrix and $1 \leq k \leq \min(m,n)$. Let $f > 1$ and Π be a permutation matrix such that the decomposition described in (1) verifies for all $(i,j) \in [1,k] \times [1,n-k]$*

$$\gamma_j^2(R_{11}^{-1}R_{12}) + \gamma_j^2(R_{22})/\sigma_{\min}^2(R_{11}) \leq kf^2$$

Then for any $1 \leq j \leq n-k$ and $1 \leq i \leq k$

$$1 \leq \frac{\sigma_i(A)}{\sigma_i(R_{11})} \leq \sqrt{1+kf^2(n-k)}, \quad 1 \leq \frac{\sigma_j(R_{22})}{\sigma_{k+j}(A)} \leq \sqrt{1+kf^2(n-k)} \tag{8}$$

We note that the lower bounds of 1 are satisfied for any permutation Π through the interlacing property of singular values.

2.3 Tournament pivoting for 1D block column partitioned matrices

In the context of computing a low rank approximation of a matrix while also minimizing communication, a communication avoiding version of strong RRQR factorization is introduced in [7]. It relies on a technique referred to as tournament pivoting for selecting k columns from the columns of the input matrix A, which proceeds as following. Consider that the matrix A is partitioned into 4 column blocks, $A = \begin{bmatrix} A_{11} & A_{12} & A_{13} & A_{14} \end{bmatrix}$. From each column block $A_{1i}, i = 1, \ldots, 4$, k columns are selected by using strong RRQR, and their indices are given in I_{i0}.

$$\begin{array}{cccc} [\ A_{11} & A_{12} & A_{13} & A_{14}\ ] \\ \| & \| & \| & \| \\ [\ Q_{00}R_{00}\Pi_{00}^T & Q_{10}R_{10}\Pi_{10}^T & Q_{20}R_{20}\Pi_{20}^T & Q_{30}R_{30}\Pi_{30}^T\ ] \\ \downarrow & \downarrow & \downarrow & \downarrow \\ I_{00} & I_{10} & I_{20} & I_{30} \end{array}$$

From these 4 sets of columns, the final k columns are selected through a reduce like operation, with the operator being the strong RRQR factorization. Thus 2 sets of k columns are concatenated, and a new set of k columns is selected with strong RRQR.

$$\begin{array}{cc} [\ A[:, I_{00} \cup I_{10}] & A[:, I_{20} \cup I_{30}];\ ] \\ \| & \| \\ [\ Q_{01}R_{01}\Pi_{01}^T & Q_{11}R_{11}\Pi_{11}^T\ ] \\ \downarrow & \downarrow \\ I_{01} & I_{11} \end{array}$$

The final set of k columns is obtained by concatenating the two sets of k columns whose indices are in I_{01} and I_{11}, and performing strong RRQR on the obtained matrix.

$$A[:, I_{01} \cup I_{11}] = Q_{02}R_{02}\Pi_{02}^T \ \rightarrow I_{02}$$

The indices of the selected columns are in I_{02}.

In the original algorithm, each step of the reduction operation is performed on 2 sets of columns. We extend now the analysis to the case where each reduction involves a number of p sets of columns, as presented in Algorithm 1.

theorem 3 generalizes Lemma 2.5 from [7] to any number p of matrices from which k columns are selected.

Algorithm 1 1Dc-TP: Tournament pivoting for 1D column partitioned matrices (one reduction step, selection of k columns)

Require: Input matrices $A_1, \ldots, A_p$, approximation rank k
1: For each A_i, compute strong RRQR to select k columns, store indices in I_i
2: Concatenate selected columns $\bar{A} = \begin{bmatrix} A_1[:, I_1] & \cdots & A_p[:, I_p] \end{bmatrix}$
3: Compute strong RRQR of $\bar{A}$ to select k columns
Ensure: indices of k rank revealing columns of A

Theorem 3. *Let p be any strictly positive integer. Let $f > 1$ and $1 \leq k \leq \min(m, n, p)$. For any $1 \leq \ell \leq p$, let $A_\ell \in \mathbb{R}^{m \times n}$ and $f_\ell > 1$. We note $A = \begin{bmatrix} A_1 & \cdots & A_p \end{bmatrix}$ and the strong RRQR decomposition of any A_ℓ as*

$$A_\ell \Pi_\ell = Q_\ell \begin{bmatrix} R_{11}^\ell & R_{12}^\ell \\ & R_{22}^\ell \end{bmatrix},$$

where $R_{11}^\ell \in \mathbb{R}^{k \times k}$ and where Π_ℓ is such that for any $1 \leq j \leq n - k$

$$\gamma_j^2({R_{11}^\ell}^{-1} R_{12}^\ell) + \gamma_j^2(R_{22}^\ell)/\sigma_{\min}^2(R_{11}^\ell) \leq k f_\ell^2.$$

Then the horizontal concatenation $\tilde{A} = \begin{bmatrix} Q_1 \begin{bmatrix} R_{11}^1 \\ \end{bmatrix} & \cdots & Q_p \begin{bmatrix} R_{11}^p \\ \end{bmatrix} \end{bmatrix}$ factorized as
$\tilde{Q} \begin{bmatrix} \tilde{R}_{11} & \bar{R}_{12} \\ & \bar{R}_{22} \end{bmatrix}$ *with $\tilde{R}_{11} \in \mathbb{R}^{k \times k}$, such that $\left(\tilde{R}_{11}^{-1} \bar{R}_{12}\right)_{ij}^2 + \omega_i^2(\tilde{R}_{11}) \gamma_j^2(\bar{R}_{22}) \leq f^2$, verifies*

$$\gamma_j^2(\tilde{R}_{11}^{-1} \tilde{R}_{12}) + \gamma_j^2(\tilde{R}_{22})/\sigma_{\min}^2(\tilde{R}_{11}) \leq 2k^3 f^2 \max(f_1^2, \ldots, f_p^2).$$

Here $\tilde{R}_{12}$ and $\tilde{R}_{22}$ are such that $A\tilde{\Pi} = \tilde{Q} \begin{bmatrix} \tilde{R}_{11} & \tilde{R}_{12} \\ & \tilde{R}_{22} \end{bmatrix}$ and $\tilde{\Pi}$ is a permutation matrix.

Proof. Let $p > 1$. Consider Section 2.3.1 of [7] which proves the theorem for $p = 2$. We now give the main differences to generalize it for $p > 2$. Start with p matrices $A_1 \ldots A_p$ and $\tilde{A} = \begin{bmatrix} A_1 & \cdots & A_p \end{bmatrix}$. For $1 \leq \ell \leq p$ we define for A_ℓ the following matrices the same ways $\mathcal{N}$, $\mathcal{C}$, N, C_1 and C_2 are defined for B and $\hat{B}$ in section 2.3.1 of [7].

$$\begin{aligned} N &= \left(R_{11}^\ell\right)^{-1} R_{12}^\ell \\ \tilde{N} &= \tilde{R}_{11}^{-1} \bar{R}_{12} \\ \mathcal{N}_\ell &= \begin{bmatrix} \mathcal{I}_k & \tilde{N} \end{bmatrix} \bar{\Pi}^T[:, 1:k] \\ \mathcal{C}_\ell &= \begin{bmatrix} \mathcal{O} & \bar{R}_{22} \end{bmatrix} \bar{\Pi}^T[:, 1:k] \\ \begin{bmatrix} C_1^\ell \\ C_2^\ell \end{bmatrix} &= \tilde{Q}_\ell^T Q_\ell \begin{bmatrix} \\ R_{22}^\ell \end{bmatrix} \end{aligned}$$

With the same reasoning as in [7] we obtain

$$\begin{aligned} \tilde{R}_{12} &= \begin{bmatrix} \bar{R}_{12} & \tilde{R}_{11}(\mathcal{N}_1 N_1 + \tilde{R}_{11}^{-1} C_1^1) & \cdots & R_{11}(\mathcal{N}_p N_p + \tilde{R}_{11}^{-1} C_1^p) \end{bmatrix}, \\ \tilde{R}_{22} &= \begin{bmatrix} \bar{R}_{22} & \mathcal{C}_1 N_1 + C_2^1 & \cdots & \mathcal{C}_p N_p + C_2^p \end{bmatrix}. \end{aligned}$$

By adapting the second part of the proof of [7] we obtain for $1 \leq \ell \leq p$,

$$\gamma_j^2(\mathcal{N}_\ell N_\ell + R_{11}^{-1} C_1^\ell) + \gamma_j^2(\mathcal{C}_\ell N_\ell + C_2^\ell)/\sigma_{\min}^2(R_{11}) < 2f^2 k^3 f_\ell^2.$$

By combining these two relations, we obtain the result of the theorem. □

3 Tournament pivoting for 1D block row partitioned matrices

In this section we present a new algorithm to compute a spectrum preserving and kernel approximation factorization of a block row partitioned matrix by using tournament pivoting. We use this algorithm in section 4 to extend tournament pivoting to the case when a matrix is distributed over a set of processors by using a 2D partitioning of both rows and columns.

Tournament pivoting for 1D block row partitioned matrices, referred to as 1Dr-TP, selects k columns of a matrix from the selections performed on its blocks of rows using a reduction tree. It has thus similarities with the column partitioned version (1Dc-TP cf. section 2.3), however now the selections are performed from subcolumns of the matrix. We present the algebra of 1Dr-TP by using a simple example in which A is partitioned into 4 blocks of rows, $A = [A_{11}; \ldots; A_{41}]$. First k columns are selected from each block of rows $A_{i1}, i = 1, \ldots, 4$ and their indices are stored in $I_{i0}, i = 1, \ldots, 4$,

$$A = \begin{pmatrix} A_{11} \\ A_{21} \\ A_{31} \\ A_{41} \end{pmatrix} = \begin{pmatrix} Q_{00}R_{00}\Pi_{00}^{-1} & \rightarrow \text{select k cols } I_{00} \\ Q_{10}R_{10}\Pi_{10}^{-1} & \rightarrow \text{select k cols } I_{10} \\ Q_{20}R_{20}\Pi_{20}^{-1} & \rightarrow \text{select k cols } I_{20} \\ Q_{30}R_{30}\Pi_{30}^{-1} & \rightarrow \text{select k cols } I_{30} \end{pmatrix}$$

Then the sets of k indices are combined two by two to select each time a new set of k indices. For example for the first two sets, I_{00} and I_{10}, the selection is performed as following. The columns of the first two block rows $[A_{11}; A_{21}]$ whose indices belong to $I_{00} \cup I_{10}$ are concatenated together to form a new matrix, $[A_{11}; A_{21}][:, I_{00} \cup I_{10}]$. Strong RRQR is applied to this matrix to select a new set of its columns I_{01},

$$\left[\begin{array}{c} \begin{bmatrix} A_{11} \\ A_{21} \end{bmatrix}[:, I_{00} \cup I_{10}] \\ \hline \begin{bmatrix} A_{31} \\ A_{41} \end{bmatrix}[:, I_{20} \cup I_{30}] \end{array}\right] = \begin{bmatrix} Q_{01}R_{01}\Pi_{01}^{-1} & \rightarrow I_{01} \\ Q_{11}R_{11}\Pi_{11}^{-1} & \rightarrow I_{11} \end{bmatrix}$$

In the last step, the columns of A whose indices belong to $I_{20} \cup I_{30}$ are concatenated together, and the final k columns are selected through strong RRQR from $A[:, I_{01} \cup I_{11}]$,

$$A[:, I_{01} \cup I_{11}] = Q_{02}R_{02}\Pi_{02}^{-1} \rightarrow I_{02}$$

Algorithm 2 describes one reduction of 1Dr-TP, i.e. selects k columns from $A = [A_1; \ldots; A_p]$ through local selections of k indices from subcolumns of A in $A_1, \ldots, A_p$.

Algorithm 2 1Dr-TP: Tournament pivoting for row partitioned matrix (one reduction step, selection of k columns)

Require: Input matrices $A_1 \ldots A_p$, rank of approximation b
1: For each A_i, compute strong RRQR to select k columns, store indices in I_i
2: Concatenate selected columns $\bar{A} = \begin{bmatrix} A[:, I_1] & \cdots & A[:, I_p] \end{bmatrix}$
3: Compute strong RRQR of $\bar{A}$ to select k columns
Ensure: indices of k rank revealing columns of A

Similarly to theorem 3 we derive bounds on the quality of the approximation in the sense of theorem 1 from bounds corresponding to the selection at each step of the tournament.

Theorem 4. *Let p be any strictly positive integer. Let $f > 1$ and $1 \leq k \leq \min(m, n/p)$. For any $1 \leq \ell \leq p$, let $A_\ell \in \mathbb{R}^{m \times n}$ and $f_\ell > 1$. We note $A = \begin{bmatrix} A_1; \ldots; A_p \end{bmatrix}$ and the strong RRQR decomposition of any A_ℓ as,*

$$A_\ell \Pi_\ell = Q_\ell \begin{bmatrix} R_{1,\ell}^\ell & C_\ell \\ & D_\ell \end{bmatrix}, \tag{9}$$

where $R_{1,\ell}^\ell \in \mathbb{R}^{k \times k}$ and where Π_ℓ is such that for any $1 \leq j \leq n - b$,

$$\gamma_j^2({R_{1,\ell}^\ell}^{-1} C_\ell) + \gamma_j^2(D_\ell)/\sigma_{\min}^2(R_{1,\ell}^\ell) \leq k f_\ell^2. \tag{10}$$

Then the horizontal concatenation $\bar{A} = \begin{bmatrix} A\Pi_1[:, 1:k] & \cdots & A\Pi_p[:, 1:k] \end{bmatrix}$ *factorized as* $\tilde{Q} \begin{bmatrix} \tilde{R}_{11} & \bar{R}_{12} \\ & \bar{R}_{22} \end{bmatrix}$ *with* $\tilde{R}_{11} \in \mathbb{R}^{k \times k}$*, such that* $\left(\tilde{R}_{11}^{-1}\bar{R}_{12}\right)_{ij}^2 + \omega_i^2(\tilde{R}_{11})\gamma_j^2(\bar{R}_{22}) \leq f^2$*, verifies*

$$\gamma_j^2(\tilde{R}_{11}^{-1}\tilde{R}_{12}) + \gamma_j^2(\tilde{R}_{22})/\sigma_{\min}^2(\tilde{R}_{11}) \leq 2k^3 f^2 \sum_{i=1}^{p} f_i^2,$$

where $\tilde{R}_{12}$ *and* $\tilde{R}_{22}$ *are such that* $A\tilde{\Pi} = \tilde{Q} \begin{bmatrix} \tilde{R}_{11} & \tilde{R}_{12} \\ & \tilde{R}_{22} \end{bmatrix}$ *and* $\tilde{\Pi}$ *is a permutation matrix.*

Proof. Under the same hypothesis as theorem 4 and Π a permutation matrix such that

$$\Pi = \begin{bmatrix} \Pi_1[:, 1:k] & \cdots & \Pi_p[:, 1:k] & \Pi_{p+1} \end{bmatrix},$$

where Π_{p+1} selects the remaining columns so that Π is orthogonal. For the sake of clarity, blocks are supposed to reveal non overlapping sets of columns, *i.e.* $\forall \ell' \neq \ell, \Pi_\ell[:, 1:k]^T \Pi_{\ell'}[:, 1:k] = \mathcal{O}$. Using Π we can rewrite the strong rank revealing factorization (9) as,

$$A_\ell \Pi = Q_\ell \begin{bmatrix} R_{1,1}^\ell & \cdots & R_{1,p}^\ell & R_{1,p+1}^\ell \\ R_{2,1}^\ell & \cdots & R_{2,p}^\ell & R_{2,p+1}^\ell \end{bmatrix},$$

with $R_{2,\ell}^\ell = \mathcal{O}$, $R_{1,1}^\ell, \ldots, R_{1,p}^\ell \in \mathbb{R}^{k \times k}$, $R_{2,1}^\ell, \ldots, R_{2,p}^\ell \in \mathbb{R}^{(m-k) \times k}$, $R_{1,p+1}^\ell \in \mathbb{R}^{k \times (n-kp)}$ and $R_{2,p+1}^\ell \in \mathbb{R}^{(m-k) \times (n-kp)}$.

The condition (10) on the factorization of each A_ℓ can be reformulated with the alternative permutation Π as the following. For $1 \leq \ell' \leq p+1$ so that $\ell' \neq \ell$ and j is a column index of $R_{1,\ell'}^\ell$,

$$\gamma_j^2({R_{1,\ell}^\ell}^{-1} R_{1,\ell'}^\ell) + \gamma_j^2(R_{2,\ell'}^\ell)/\sigma_{\min}^2(R_{1,\ell}^\ell) \leq k f_\ell^2.$$

Recalling the hypothesis of theorem 4, $\bar{A} = A \begin{bmatrix} \Pi_1[:, 1:k] & \cdots & \Pi_p[:, 1:k] \end{bmatrix} = A\Pi[:, 1:kp] \in \mathbb{R}^{mp \times kp}$ contains the columns of A selected on each block A_ℓ.

The RRQR factorization of $\bar{A}$ is defined as,

$$\bar{A}\bar{\Pi} = \tilde{Q} \begin{bmatrix} \tilde{R}_{11} & \bar{R}_{12} \\ & \bar{R}_{22} \end{bmatrix},$$

where $\tilde{Q} \in \mathbb{R}^{mp \times mp}$ and $\tilde{R}_{11} \in \mathbb{R}^{k \times k}$ such that for any $1 \leq i \leq k$ and $1 \leq j \leq k(p-1)$

$$\left(\tilde{R}_{11}^{-1}\bar{R}_{12}\right)_{ij}^2 + \omega_i^2(\tilde{R}_{11})\gamma_j^2(\bar{R}_{22}) \leq f^2 \tag{11}$$

Finally, the global approximation can be expressed as

$$A\tilde{\Pi} = \tilde{Q} \begin{bmatrix} \begin{bmatrix} \tilde{R}_{11} & \bar{R}_{12} \\ & \bar{R}_{22} \end{bmatrix}, & \tilde{Q}^T \begin{bmatrix} Q_1 \begin{bmatrix} R_{1,p+1}^1 \\ R_{2,p+1}^1 \end{bmatrix} \\ \vdots \\ Q_p \begin{bmatrix} R_{1,p+1}^p \\ R_{2,p+1}^p \end{bmatrix} \end{bmatrix} \end{bmatrix} =_{def} \tilde{Q} \begin{bmatrix} \tilde{R}_{11} & \tilde{R}_{12} \\ & \tilde{R}_{22} \end{bmatrix}$$

We will now express $\tilde{R}_{12}$ and $\tilde{R}_{22}$. Let $\tilde{Q} = \begin{bmatrix} \tilde{Q}_1; \ldots; \tilde{Q}_p \end{bmatrix}$, where $\forall \ell \leq p, \tilde{Q}_\ell \in \mathbb{R}^{m \times mp}$ leading to,

$$\tilde{Q}^T \begin{bmatrix} Q_1 \begin{bmatrix} R_{1,p+1}^1 \\ R_{2,p+1}^1 \end{bmatrix} \\ \vdots \\ Q_p \begin{bmatrix} R_{1,p+1}^p \\ R_{2,p+1}^p \end{bmatrix} \end{bmatrix} = \sum_{\ell=1}^{p} \tilde{Q}_\ell^T Q_\ell \begin{bmatrix} R_{1,p+1}^\ell \\ R_{2,p+1}^\ell \end{bmatrix}.$$

We can express, with $N_\ell = {R^\ell_{1,\ell}}^{-1} R^\ell_{1,p+1}$,

$$\tilde{Q}_\ell^T Q_\ell \begin{bmatrix} R^\ell_{1,p+1} \\ R^\ell_{2,p+1} \end{bmatrix} = \tilde{Q}_\ell^T Q_\ell \begin{bmatrix} R^\ell_{1,\ell} \\ \end{bmatrix} N_\ell + \tilde{Q}_\ell^T Q_\ell \begin{bmatrix} \\ R^\ell_{2,p+1} \end{bmatrix} \tag{12}$$

To ease the selection notation, let $I_\ell = [(\ell-1)k+1 : \ell k]$ be the indices of the columns of $\bar{A}$ corresponding to the selected columns of A_ℓ. From the definition of $\bar{A}$,

$$\begin{aligned} Q_\ell \begin{bmatrix} R^\ell_{1,\ell} \\ \end{bmatrix} &= \tilde{Q}_\ell \begin{bmatrix} \tilde{R}_{11} & \bar{R}_{12} \\ & \bar{R}_{22} \end{bmatrix} \bar{\Pi}^T[:, I_\ell] \\ &= \tilde{Q}_\ell \begin{bmatrix} \tilde{R}_{11}\mathcal{N}_\ell \\ \mathcal{C}_\ell \end{bmatrix}, \end{aligned}$$

with

$$\begin{aligned} \mathcal{N}_\ell &= \begin{bmatrix} \mathcal{I}_k & \tilde{N} \end{bmatrix} \bar{\Pi}^T[:, I_\ell], \\ \mathcal{C}_\ell &= \begin{bmatrix} \mathcal{O} & \bar{R}_{22} \end{bmatrix} \bar{\Pi}^T[:, I_\ell], \\ \tilde{N} &= \tilde{R}_{11}^{-1} \bar{R}_{12}. \end{aligned}$$

Note that the new introduced matrices verify for $1 \le i \le k$ and $1 \le j \le k$,

$$(\mathcal{N}_\ell)^2_{i,j} + \gamma_j^2(\mathcal{C}_\ell)\omega_i^2(\tilde{R}_{11}) \le f^2. \tag{13}$$

We also define,

$$\tilde{Q}_\ell^T Q_\ell \begin{bmatrix} \\ R^\ell_{2,p+1} \end{bmatrix} =_{def} \begin{bmatrix} C^\ell_1 \\ C^\ell_2 \end{bmatrix}.$$

Plugging everything using the global approximation expression, we obtain,

$$\tilde{Q}^T \begin{bmatrix} Q_1 \begin{bmatrix} R^1_{1,p+1} \\ R^1_{2,p+1} \end{bmatrix} \\ \vdots \\ Q_p \begin{bmatrix} R^p_{1,p+1} \\ R^p_{2,p+1} \end{bmatrix} \end{bmatrix} = \sum_{\ell=1}^{p} \left(\begin{bmatrix} \tilde{R}_{11}\mathcal{N}_\ell \\ \mathcal{C}_\ell \end{bmatrix} N_\ell + \begin{bmatrix} C^\ell_1 \\ C^\ell_1 \end{bmatrix} \right),$$

and then

$$\begin{aligned} \tilde{R}_{12} &= \begin{bmatrix} \bar{R}_{12} & \tilde{R}_{11} \sum_{\ell=1}^{p} \left(\mathcal{N}_\ell N_\ell + \tilde{R}_{11}^{-1} C_1^\ell \right) \end{bmatrix}, \\ \tilde{R}_{22} &= \begin{bmatrix} \bar{R}_{22} & \sum_{\ell=1}^{p} \left(\mathcal{C}_\ell N_\ell + C_2^\ell \right) \end{bmatrix}. \end{aligned}$$

We are looking for an upper bound of the following expression

$$\gamma_j^2(\tilde{R}_{11}^{-1}\tilde{R}_{12}) + \gamma_j^2(\tilde{R}_{22}) / \sigma^2_{\min}(\tilde{R}_{11}). \tag{14}$$

Case 1 If $1 \le j \le k$, then $\gamma_j^2(\tilde{R}_{11}^{-1}\tilde{R}_{12}) = \gamma_j^2(\tilde{R}_{11}^{-1}\bar{R}_{12})$ and $\gamma_j^2(\tilde{R}_{22}) = \gamma_j^2(\bar{R}_{22})$. So we can use (11) which gives (14) $< kf^2$.

Case 2 Assume $k+1 \leq j \leq n-kp$. We have,

$$\begin{aligned}(14) &= \gamma_j^2\left(\sum_{\ell=1}^{p}\left(\mathcal{N}_\ell N_\ell + \tilde{R}_{11}^{-1}C_1^\ell\right)\right) + \gamma_j^2\left(\sum_{\ell=1}^{p}\left(\mathcal{C}_\ell N_\ell + C_2^\ell\right)/\sigma_{\min}^2(\tilde{R}_{11})\right)\\ &= \gamma_j^2\left(\sum_{\ell=1}^{p}\left(\begin{bmatrix}\mathcal{N}_\ell N_\ell\\ \mathcal{C}_\ell N_\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix} + \begin{bmatrix}\tilde{R}_{11}^{-1}C_1^\ell\\ C_2^\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix}\right)\right)\\ &\leq 2p\sum_{\ell=1}^{p}\left(\gamma_j^2\left(\begin{bmatrix}\mathcal{N}_\ell N_\ell\\ \mathcal{C}_\ell N_\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix}\right) + \gamma_j^2\left(\begin{bmatrix}\tilde{R}_{11}^{-1}C_1^\ell\\ C_2^\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix}\right)\right).\end{aligned}$$

On one hand we can say using the relaxed form (cf. (7)) of (13) that for any $1 \leq \ell \leq p$,

$$\gamma_j^2\left(\begin{bmatrix}\mathcal{N}_\ell N_\ell\\ \mathcal{C}_\ell N_\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix}\right) \leq \left\|\begin{bmatrix}\mathcal{N}_\ell\\ \mathcal{C}_\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix}\right\|_F^2 \gamma_j^2(N_\ell) \leq f^2k^2\gamma_j^2(N_\ell). \tag{15}$$

On other hand using in a similar manner as (15) that for any $x \in \mathbb{R}^k$, $\|\tilde{R}_{11}^{-1}x\| \leq \sigma_{\max}(\tilde{R}_{11}^{-1})\|x\| = \sigma_{\min}(\tilde{R}_{11})\|x\|$,

$$\gamma_j^2\left(\begin{bmatrix}\tilde{R}_{11}^{-1}C_1^\ell\\ C_2^\ell/\sigma_{\min}(\tilde{R}_{11})\end{bmatrix}\right) \leq \gamma_j^2\left(\begin{bmatrix}C_1^\ell\\ C_2^\ell\end{bmatrix}\right)/\sigma_{\min}^2(\tilde{R}_{11}).$$

Furthermore using definition of C_1^ℓ and C_2^ℓ,

$$\sum_{\ell=1}^{p}\gamma_j^2\left(\begin{bmatrix}C_1^\ell\\ C_2^\ell\end{bmatrix}\right) = \gamma_j^2\left(\begin{bmatrix}\begin{bmatrix}C_{1,1}^1\\ C_{2,1}^1\end{bmatrix}\\ \vdots\\ \begin{bmatrix}C_{1,p}^p\\ C_{2,p}^p\end{bmatrix}\end{bmatrix}\right) = \gamma_j^2\left(\begin{bmatrix}R_{2,p+1}^1\\ \vdots\\ R_{2,p+1}^p\end{bmatrix}\right) = \sum_{\ell=1}^{p}\gamma_j^2\left(R_{2,p+1}^\ell\right). \tag{16}$$

We can use relations (15) and (16) to derive further (14).

$$(14) \leq 2p\sum_{\ell=1}^{p}\left(f^2k^2\gamma_j^2(N_\ell) + \frac{\gamma_j^2(R_{2,p+1}^\ell)}{\sigma_{\min}^2(\tilde{R}_{11})}\right).$$

Firstly considering that for any $1 \leq \ell \leq p$,

$$\sigma_{\min}^2\left(\begin{bmatrix}\tilde{R}_{11}\mathcal{N}_\ell\\ \mathcal{C}_\ell\end{bmatrix}\right) \leq \sigma_{\min}^2\left(\tilde{R}_{11}\right)\|\mathcal{N}_\ell\|_2^2 + \|\mathcal{C}_\ell\|_2^2 \leq f^2k^2\sigma_{\min}^2(\tilde{R}_{11}),$$

secondly with the interlacing singular values theorem,

$$\sigma_{\min}^2(R_{1,\ell}^\ell) \leq \sigma_{\min}^2\left(\bar{A}[:, I_\ell]\right),$$

and the fact that

$$\begin{bmatrix}Q_1 & & \\ & \ddots & \\ & & \hat{Q}_p\end{bmatrix} A[:, I_\ell] = \tilde{Q}\begin{bmatrix}\tilde{R}_{11}\mathcal{N}_\ell\\ \mathcal{C}_\ell\end{bmatrix},$$

we obtain

$$\sigma_{\min}^2(R_{1,\ell}^\ell) \leq f^2k^2\sigma_{\min}^2(\tilde{R}_{11}).$$

Using these two relations we continue to derive (14),

$$\begin{aligned}(14) &\leq 2pf^2k^2\sum_{\ell=1}^{p}\left(\gamma_j^2(N_\ell) + \frac{\gamma_j^2(R_{2,p+1}^\ell)}{\sigma_{\min}^2(R_{1,\ell}^\ell)}\right)\\ &\leq 2pf^2k^3\sum_{\ell=1}^{p}f_\ell^2.\end{aligned}$$

□

4 QR factorization with 2D tournament pivoting

In this section we introduce an algorithm for computing a low rank approximation of a matrix distributed over a 2D grid of processors by using tournament pivoting. We refer to this pivoting strategy as 2D tournament pivoting, or 2D TP, which combines the two pivoting strategies introduced in the previous sections, 1Dc-TP for matrices partitioned into blocks of columns and 1Dr-TP for matrices partitioned into blocks of rows.

4.1 QRTP algorithm

We consider a matrix $A \in \mathbb{R}^{m\times n}$ distributed on a $P_r \times P_c$ grid of processors. We explain the algorithm considering a 2×4 grid of processors, that is A is partitioned as,

$$A = \begin{pmatrix} A_{11} & A_{12} & A_{13} & A_{14} \\ A_{21} & A_{22} & A_{23} & A_{24} \end{pmatrix}.$$

We consider here that tournament pivoting relies on a binary tree. The general case will be covered in a following section. First, k columns are selected from each column block by using binary 1Dr-TP.

$$\begin{array}{cccc} \begin{pmatrix} A_{11} \\ A_{21} \end{pmatrix} & \begin{pmatrix} A_{12} \\ A_{22} \end{pmatrix} & \begin{pmatrix} A_{13} \\ A_{23} \end{pmatrix} & \begin{pmatrix} A_{14} \\ A_{24} \end{pmatrix} \\ \downarrow & \downarrow & \downarrow & \downarrow \\ I_{00} & I_{10} & I_{20} & I_{30} \end{array}$$

Second, binary 1Dc-TP is applied on the sets of k selected columns to obtain the final k columns.

$$\begin{array}{cccc} A(:, I_{00}) & A(:, I_{10}) & A(:, I_{20}) & A(:, I_{30}) \\ & \downarrow & & \\ & I_{02} & & \end{array}$$

Figure 2 illustrates this algebra and algorithm 3 gives the formal procedure for computing the QR factorization of a matrix with 2D tournament pivoting, referred to as QRTP.

Algorithm 3 QRTP: QR factorization with 2D tournament pivoting

Require: $P_r \times P_c$ processor grid, A distributed matrix, k approximation rank
1: **for** $1 \le i \le P_r$ in parallel **do**
2: $\quad I_i \leftarrow$ select k columns from each block column i using binary 1Dr-TP
3: **end for**
4: $I \leftarrow$ Select k columns from $A(:, \cup_{i=1}^{P_r} I_i)$ using binary 1Dc-TP
Ensure: I indices of k rank revealing columns of A

We now give the bounds for the approximations of the singular values obtained by QRTP. We assume that each selection of k columns in the algorithm is performed such that the bounds in (6) are satisfied with a constant f. Let $P = P_r P_c$. Then with

$$f_{TP} = \sqrt{P} P_r k^{\log_2(P)}\, f^{\log_2(P)+1}$$

and by using theorem 2, it can be shown that the singular values of $\tilde{R}_{11}$ and $\tilde{R}_{22}$ approximate the singular values of A in the following sense: for $1 \le j \le n-k$ and $1 \le i \le k$,

$$1 \le \frac{\sigma_i(A)}{\sigma_i(\tilde{R}_{11})} \le \sqrt{1 + k f_{TP}^2 (n-k)}, 1 \le \frac{\sigma_j(\tilde{R}_{22})}{\sigma_{j+k}(A)} \le \sqrt{1 + k f_{TP}^2 (n-k)}.$$

The steps to find these bounds are detailed in section 4.2.2 in a more general setting.

4.2 Spectrum preserving and kernel approximation properties of QRTP

We discuss in this section the spectrum preserving and kernel approximation properties of QRTP by considering more general reduction trees. We also compare the reduction strategy used in QRTP with an alternative one in terms of approximation bounds.

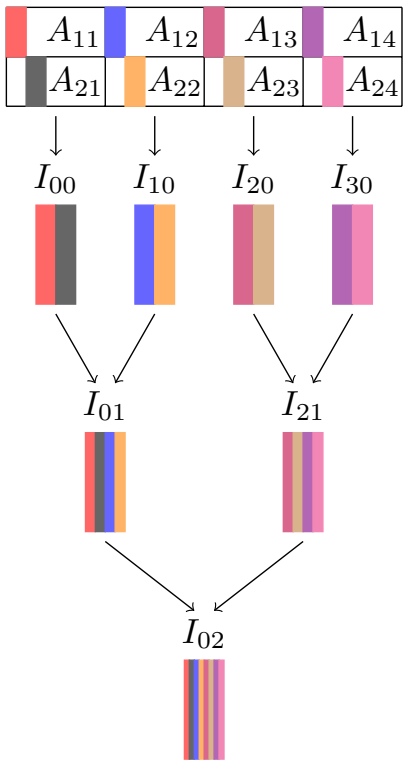

Figure 2: Column selection using algorithm 3. A set of k columns are selected for each submatrix A_{ij}, then these columns are combined along a reduction tree to find k columns for the complete matrix A. I represents a set of indices.

4.2.1 Spectrum preserving and kernel approximation properties of 1Dc-TP and 1Dr-TP

We discuss first these two properties for both 1Dc-TP and 1Dr-TP. Let A be partitioned into P_c blocks of columns and let A_i be the i-th block. Let k be the rank of all approximations. Consider that the selection of k columns from each block A_i satisfies inequality (7) with the upper bound being F_i. Consider the h^{th} reduction step out of D reduction steps of the algorithm and let f_i^h be the bound corresponding to the selection of k columns at this reduction step, and let x_h be the number of column sets implied in this reduction. Using induction on theorem 3, we obtain that the selection of k columns from A leads to a factorization based on 1Dc-TP that satisfies (7) with the upper bound being,

$$F_c^2 = (2k^2)^D \max_{1\leq i\leq P_c} \left(F_i^2 \prod_{h=1}^{D} (f_i^h)^2 \right). \tag{17}$$

This expression relates to Corollary 2.6 and 2.7 in [7] by assuming that all approximations enforce the same bound f in inequality (7), i.e. for $1 \leq i \leq P_c$ and $1 \leq h \leq D$, $F_i = \sqrt{k} f$ and $f_i^h = f$.

Now assume that A is partitioned into P_r blocks of rows. With the same conventions as for (17), using induction on theorem 4, the approximation obtained by selecting k columns from A using 1Dr-TP satisfies (7) with the upper bound being,

$$F_r^2 = (2k^2)^D \sum_{1\leq i\leq N} \left(F_i^2 \prod_{h=1}^{D} x_h (f_i^h)^2 \right). \tag{18}$$

4.2.2 Row-first and column-first strategies

When considering a matrix partitioned into $P_r \times P_c$ blocks, the choice of reduction tree impacts the accuracy and the performance of QRTP. In section 4 we described 2D TP by using first 1Dr-TP followed by 1Dc-TP, we refer to this strategy as row-first strategy. Additionally we define the column-first strategy as 1Dc-TP followed by 1Dr-TP, and we show in this section that given any partitioning of A, the row-first strategy has better theoretical approximation bounds than the column-first strategy. There can be more complex ways to combine row and column reductions, but we do not explore them in this paper.

Let A be a matrix partitioned into $P_r \times P_c$ blocks and $A_{i,j}$ be the block at indices (i, j) in the partitioned matrix. Let k be the rank of all approximations. Let $F_{i,j}$ be the bound associated with the selection of k columns from $A_{i,j}$ in the sense of (7). We assume that 1Dr-TP is composed of D_r reduction steps and 1Dc-TP has D_c reduction steps. Let $f_{i,j}^h$ be the bounds associated with the h^{th} reduction step of the partition $A_{i,j}$. Replacing each F_i in (17) with F_r from (18), we obtain the following bounds in the sense of (6) for the row-first strategy,

$$F_{rc}^2 = (2k^2)^{D_r+D_c} \max_{1\leq j\leq P_c} \left(\sum_{i=1}^{P_r} \left[F_{i,j}^2 \prod_{h=1}^{D_r+D_c} (f_{i,j}^h)^2 \prod_{h=1}^{D_r} x_i \right] \right). \tag{19}$$

Replacing each F_i in (18) with F_c from (17), we have the following bounds in the sense of (6) for the column-first strategy,

$$F_{cr}^2 = (2k^2)^{D_r+D_c} \sum_{i=1}^{P_r} \left[\max_{1\leq j\leq P_c} \left(F_{i,j}^2 \prod_{h=1}^{D_c+D_r} (f_{i,j}^h)^2 \prod_{h=D_c+1}^{D_c+D_r} x_i \right) \right]. \tag{20}$$

We conclude with the following corollary.

Corollary 1. *Let A be a matrix partitioned into $P_r \times P_c$ blocks for which a low rank approximation based on QRTP is computed using 1Dr-TP and 1Dc-TP with given reduction trees. Let F_{rc} resp. F_{cr} be the bounds in the sense of* (6) *obtained when executing QRTP with 1Dr-TP followed by 1Dc-TP resp. 1Dc-TP followed by 1Dr-TP. We have the following relation on the bounds,*

$$F_{rc} \leq F_{cr}.$$

Corollary 1 shows that once the reduction trees are fixed for 1Dr-TP and 1Dc-TP, the row-first strategy always has a smaller lower bound than the column-first strategy. In more details, it means that given a matrix partitioned into $P_r \times P_c$ blocks for which a low rank approximation is computed using QRTP, the guarantees of the low rank approximations in the sense of theorem 1 are better when reducing first along the first dimension (1Dr-TP to select P_c subsets of columns, then 1Dc-TP to select k columns) than along the second dimension (1Dc-TP to select P_r subsets of columns, then 1Dr-TP to select k columns).

5 Numerical results

In this section we study the numerical behavior of QRTP on matrices of small size. The parallel performance of the algorithm on large matrices is studied later in section 7. We consider the following two matrices,

- `heat` is a 1000×1000 matrix modeling a inverse heat equation[2],
- `gravity` is a 1000×1000 matrix modeling a gravity problem[16].

Let A be one of these two matrices and A_k its low rank approximation computed with QRTP. Figure 3 displays the first 50 singular values of the matrix `heat` and their approximations computed with QRTP considering that the matrix is partitioned into 8×8 blocks (which corresponds to executing the algorithm on 64 processors). The red dots give the ratio between the singular values of A and the singular values of the QRTP approximation, $\sigma_i(A_k)/\sigma_i(A)$. The approximations $\sigma_i(A_k)$ are computed by applying the SVD to A_k. Figure 3 shows that the QRTP approximation gives a very accurate approximation of the 40 largest singulars values of A with a relative error smaller than 2.5%. For the singular values 41 to 48, the relative error increases to 10% and reaches 20% for singular values 49 and 50. It also shows that QRTP is close to the QRCP approximation, and for some singular values QRTP is more accurate, see for example the 48th singular values.

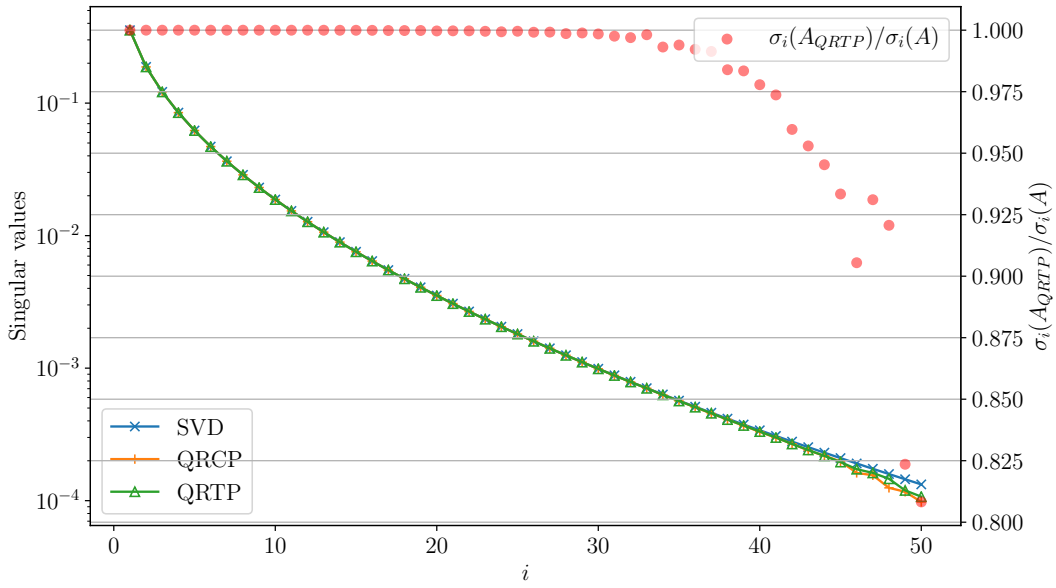

Figure 3: Singular values of the matrix `heat` and its approximation with QRCP and QRTP on 64 processors. The rank of the approximation is 50. The red dots give the ratio of the singular values of the approximation with QRTP over the original singular values.

Figure 4 presents the same analysis for the matrix `gravity`. The singular values of this matrix decrease at two different speeds, where the change in speed occurs around the 25th singular value. The first 25 largest

singular values decrease with a logarithmic slope of 0.24 power of ten per singular value, whereas the 25 next largest singular values decrease with a linear slope of $2.4.10^{-6}$ per singular values. This figure shows that the QRTP approximation gives a very accurate approximation for the 22 largest singular values of the matrix `gravity` with an error smaller than 1%. From the 23rd singular value the error increases to reach 30% for the 26th singular value and is between 25% and 42% for the singular values 27 to 50. In addition we see that the QRCP and the QRTP algorithms gives an almost identical approximation.

To further study the difference between QRCP and QRTP, let us give additional data. The relative error between QRCP and QRTP is $\frac{\|A-A_{QRTP}\|_F-\|A-A_{QRCP}\|_F}{\|A-A_{QRCP}\|_F} = -4.9 \times 10^{-5}$, showing that both algorithms have indeed very close results, with QRTP performing slightly better than QRCP in term of the Frobenius norm. Nevertheless, the projection bases are different, with only 5 columns of A selected by both QRCP and QRTP. For the matrix `heat` the relative error between QRCP and QRTP is -0.06 and there are 4 common columns to create the projection bases, leading to the same conclusion for the matrix `heat`: both approximations have very close results even though the projection bases are very different.

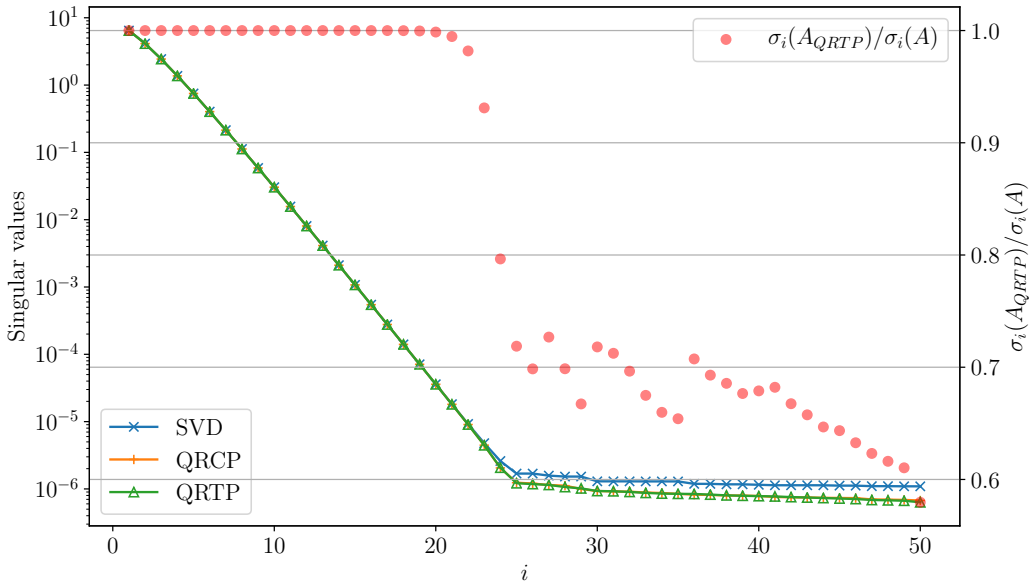

Figure 4: Singular values of the matrix `gravity` and its approximation with QRCP and QRTP on 64 processors. The rank of the approximation is 50. The red dots give the ratio of the singular values of the approximation with QRTP over the original singular values.

5.1 Influence of the reduction tree used during tournament pivoting

We study now the influence of the structure of the reduction tree on the accuracy of QRTP. We define the degree of the tree as the number of children of each node. We assume the degree is constant for a given tree, and we study the impact of this parameter along with the type of reduction executed first, 1Dr-TP versus 1Dc-TP.

We compare the following strategies for QRTP. We set the number of processors to 64 such that the matrix is partitioned into 8×8 blocks and we compute approximations of rank $k = 50$.

- Row-first of degree 2: 1Dr-TP with reduction tree of degree 2, followed by 1Dc-TP with reduction tree of degree 2. Therefore, each 1D tournament executes three reductions.
- Row-first of degree 8: 1Dr-TP with a reduction tree of degree 8, followed by 1Dc-TP with a reduction tree of degree 8. Therefore, each 1D tournament executes one reduction.

- Column-first of degree 2: 1Dc-TP with reduction tree of degree 2, followed by 1Dr-TP with reduction tree of degree 2. Therefore, each 1D tournament executes three reductions.
- Column-first of degree 8: 1Dc-TP with reduction tree of degree 8, followed by 1Dr-TP with reduction tree of degree 8. Therefore, each 1D tournament executes one reduction.

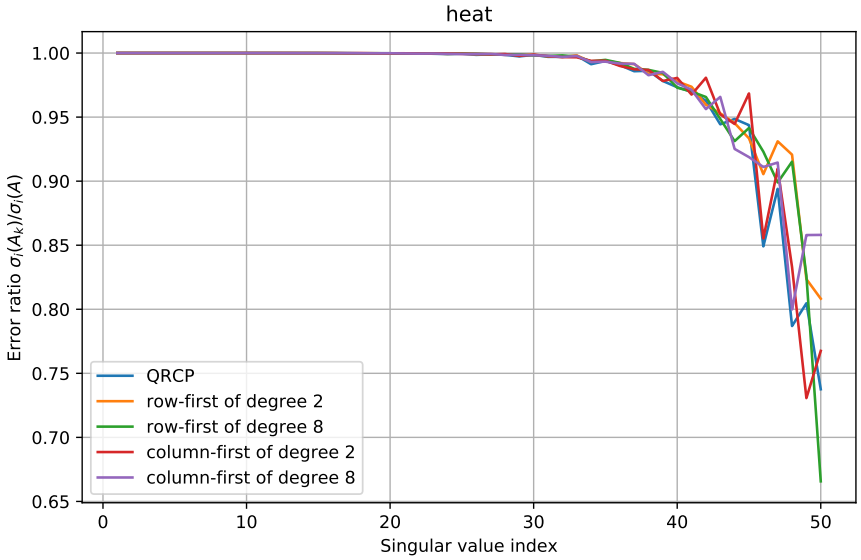

Figure 5: Error on the singular values $\sigma_i(A_k)/\sigma_i(A)$ of the matrix `heat` distributed on a 8×8 processor grid using generalized QRTP with a rank 10 and different reduction trees. Degree 2 means that the 64 blocks are combined 2 by 2, degree 8 means that the 64 blocks are combined 8 by 8. Column-first and row-first refer to the reduction strategies defined in section 4.2.2.

The results, displayed in Figure 5 for the matrix `heat` and Figure 6 for the matrix `gravity`, show the ratios of the approximated leading singular values with respect to the singular values as computed by SVD, i.e. $\frac{\sigma_i(A_k)}{\sigma_i(A)}$, where $\sigma_i(A)$ is the i-th singular value of the input matrix and $\sigma_i(A_k)$ is the singular value of the approximation computed as explained previously. In this experiment the number of cores is always 64, what changes is the strategy to combine the selected columns. Corollary 1 suggests that the row-first strategy is more accurate than the column-first strategy, and the relations (17) and (18) lead to the conclusion that a lower degree strategy is less accurate (it has indeed more intermediate reductions). We use these two claims to state the following relation on the theoretical bounds of the approximations computed with QRTP using the differents strategies.

$$F_{\text{column\_first,degree 2}} \leq F_{\text{column\_first,degree 8}}, F_{\text{row\_first,degree 2}} \leq F_{\text{row\_first,degree 8}}. \tag{21}$$

Note that the bounds in (21) are only lower bounds for the ratios of the singular values and do not give the exact precision of the algorithm, because a relation between two lower bound does not give any conclusion on the relation between the two actual values. We mention it here to merely link the theoretical and practical conclusions.

Going back to fig. 5 and fig. 6 we see that all strategies give very similar results. It invalidates in this case the expectations behind the relation (21), that there is an inequality relation between the accuracies of the different strategies. This leads to conclude that for these matrices the strategy does not impact the accuracy of QRTP, meaning that an approximation can be computed with the cheapest reduction tree without much loss of accuracy.

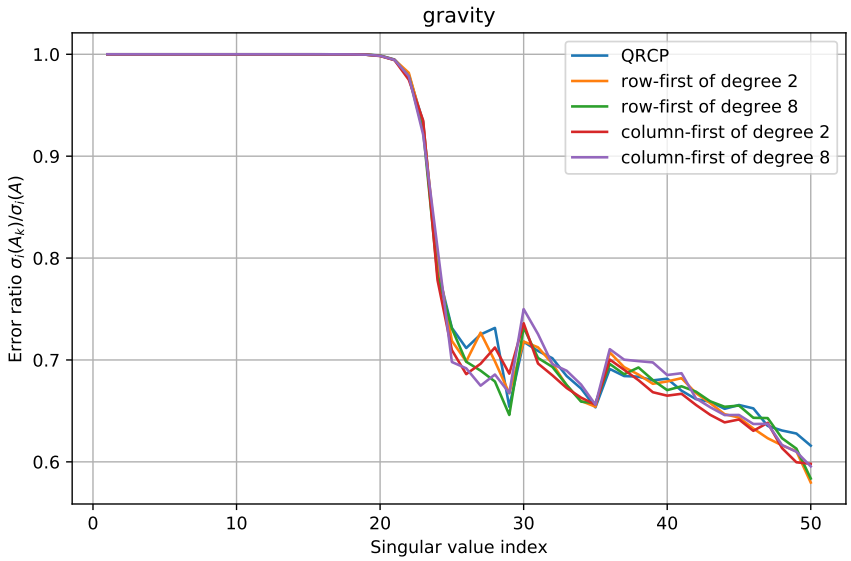

Figure 6: Error on the singular values $\sigma_i(A_k)/\sigma_i(A)$ of the matrix `gravity` distributed on a 8×8 processor grid using generalized QRTP with a rank 10 and different reduction trees. Degree 2 means that the 64 blocks are combined 2 by 2, degree 8 means that the 64 blocks are combined 8 by 8. Column-first and row-first refer to the reduction strategies defined in section 4.2.2.

5.2 QRTP for image compression

We discuss here the results obtained by QRTP when used to compress images. While we do not claim that these methods should be used for image compression, we use them to visually display and interpret the selection of columns done by tournament pivoting on partitioned matrices. We use black and white images as matrices. After applying a low rank approximation, the factors are multiplied back such that the approximation matrix can be displayed as an image and can be compared with the original image.

fig. 7 presents an image and its compressions obtained using truncated SVD, QRCP, and QRTP with a truncation rank of 10. The last image shows the 10 columns selected by QRTP. Figure 8 gives the singular values of this image and their approximations obtained by the different algorithms. Performing a rank-10 approximation brings a lot of distortion to the original image. But only few differences can be seen between the images obtained by different approximation algorithms. QRCP and QRTP have less information for the left and the right sides of the picture (note that the selected columns are mainly in the middle of the picture), leading to the presence of lines in this area. Moreover, QRCP and QRTP have very close results, showing that the subset selected by QRTP is close to the one selected by QRCP. The singular values represented in fig. 8 confirm this, with an error ratio for QRTP from 1 to 0.6. We see that the matrix corresponding to the image is close to low rank and the approximated singular values are very close to the original ones.

Additional results obtained for four other images are displayed in fig. 9, where the results of QRCP and QRTP use truncation ranks 10 and 50. We see that these images have a more complex structure than the first one, which explains the need of a higher rank to capture more details in the compressed versions. We also see that the images compressed with QRTP and QRCP are very similar.

6 Parallel design of QRTP

In this section we present the parallel design of algorithm 3 and then we determine its cost in terms of computation, number of messages and volume of communication.

The parallel implementation of QRTP is described in algorithm 4 with the following notation. For the coordinate (i, j) on the grid, A_{ij} is the submatrix of A on processor P_{ij}. Let H be the depth of the reduction

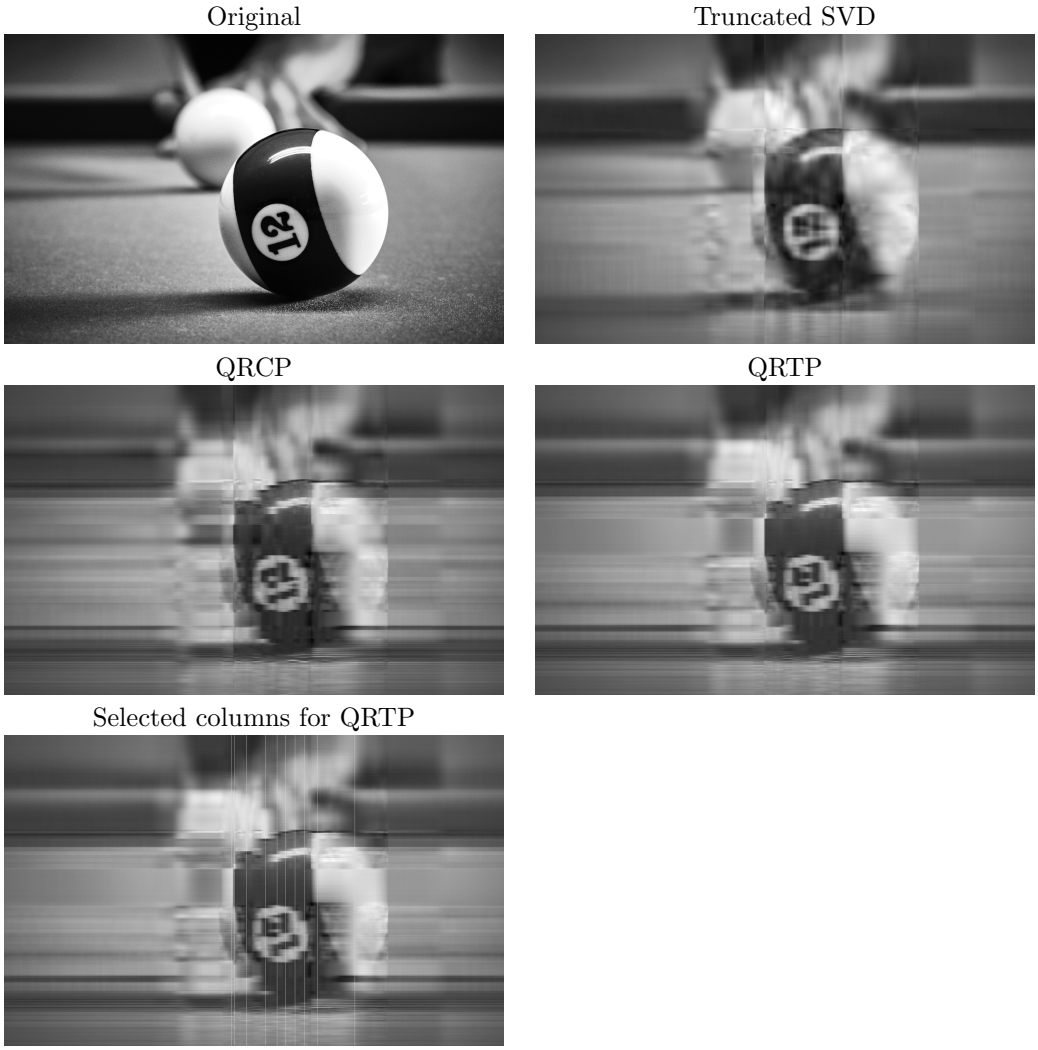

Figure 7: Compression of a 1190×1920 image for different algorithms with an approximation rank 10. QRTP uses a binary row-first reduction tree on a 8×8 processor grid.

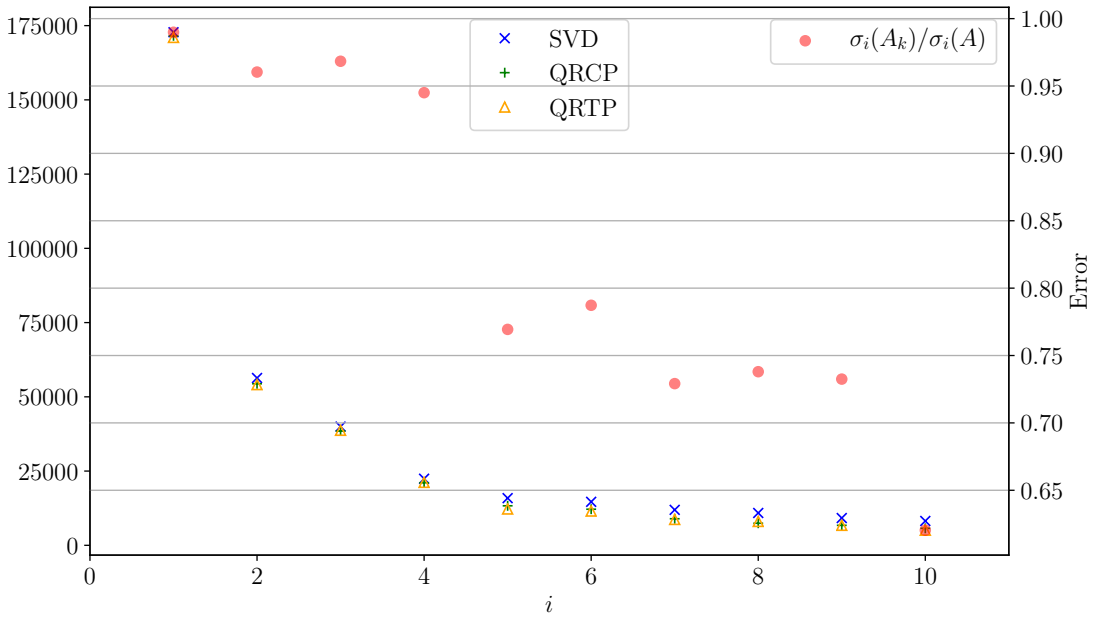

Figure 8: Singular values of the billiard ball image, original and compressed with QRCP and QRTP. The scale on the right gives the ratio $\sigma_i(A_k)/\sigma_i(A)$ where A is the matrix corresponding to the picture and A_k is its approximation with QRTP.

tree and for $0 \leq h \leq H$, I_{ij}^h is the set of columns selected by the h^{th} reduction of A_{ij} (in particular I_{ij}^0 is the set of columns selected locally for A_{ij} and I_{ij}^H is the set of final columns, output of the algorithm). Let C_{ij}^h be the MPI communicator containing processors in the subtree having root I_{ij}^h. Considering that there is a processor in each communicator elected to own the selected columns at the end of the reduction (for example MPI rank 0), let $\mathcal{C}_{ij}^h$ be the communicator containing processors $P_{i'j'}$ such that $P_{i'j'} \in C_{ij}^h$ and $P_{i'j'}$ is elected for the reduction $I_{i'j'}^{h-1}$. Figure 10 illustrates how the two kinds of communicators are composed. In algorithm 4 the processors are not descheduled during binary 1Dr-TP, corresponding to the first iterations, but half of them are idle at each iteration of binary 1Dc-TP, corresponding to the last iterations of QRTP.

We now determine the computation and communication cost of QRTP as described in algorithm 4 for a row-first strategy (binary 1Dr-TP followed by binary 1Dc-TP) and $A \in \mathbb{R}^{mP_r \times nP_c}$. The maximum computation cost per processor is,

$$\begin{aligned} \#flops(QRTP) &= \texttt{QRCP}(m \times n, k) + \log_2(P_r)\texttt{QRCP}(2k \times 2k, k) \\ &+ \sum_{i=1}^{\log_2(P_r)} \texttt{TSQR}(m \times 2k, 2^i) + \log_2(P_c)\texttt{TSQR}(m \times 2k, P_r) \\ &+ \log_2(P_c)\texttt{QRCP}(2k \times 2k, k), \end{aligned}$$

where $\texttt{QRCP}(m \times n, k)$ represents the computational cost of the rank k QRCP algorithm on an $m \times n$ matrix, which we use in practice to substitute strong RRQR, and $\texttt{TSQR}(m \times n, P_r)$ represents the cost of TSQR [5] applied on P_r blocks of size $m \times n$. Considering that $\texttt{QRCP}(m \times n, k) = 4mnk$ [10] and $\texttt{TSQR}(m \times n, P_r) = 2n^2(m + (5\log_2 P_r - 1)\frac{n}{3})$, we obtain

$$\#flops(QRTP) = 4mnk + K_1k^3 + K_2mk^2,$$

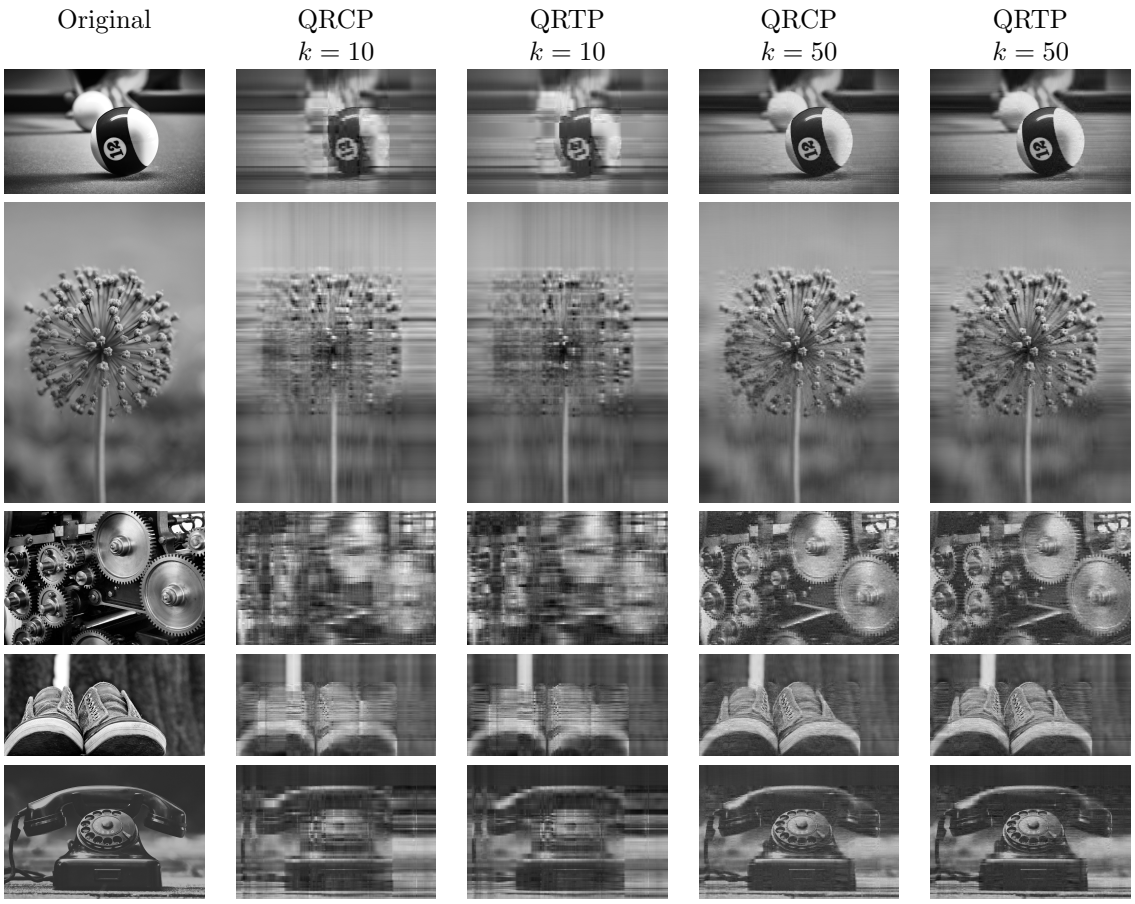

Figure 9: Compression of different images using QRCP and QRTP. QRTP uses a binary row-first division tree on a 8×8 processor grid.

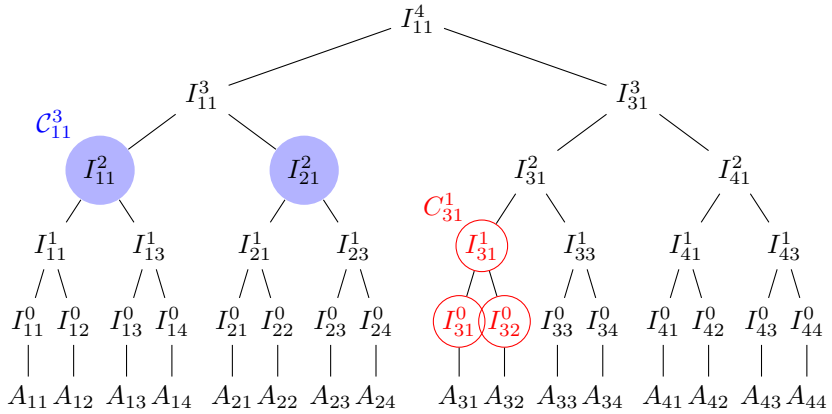

Figure 10: Reduction tree of QRTP on a 4×4 processor grid. Each node of the tree represents a reduction to select k columns from $2k$ columns concatenated from its children. Sets of processors are represented to illustrate the notation of MPI communicators. The communicator $\mathcal{C}^3_{11}$ regroups the elected processors on the children subtrees of the reduction I^3_{11} *i.e.* processors P_{11} and P_{21}. The communicator C^1_{31} regroups all processors in the subtree of reduction I^1_{31} *i.e.* P_{31} and P_{32}.

Algorithm 4 Parallel QRTP

Require: input matrix A, reduction tree S, rank of the approximation k, current processor P_{ij}

Compute strong RRQR of A_{ij} to select k columns $\rightarrow I_{ij}^0$
for h from 1 to H **do**
  if $\{A_{ij}, P_{ij} \in \mathcal{C}_{ij}^h\}$ is a row partition **then**
    if $P_{ij} \in \mathcal{C}_{ij}^h$ **then**
      Merge I_{ij}^{h-1} on communicator $\mathcal{C}_{ij}^h \rightarrow \bar{I}_{ij}^h$
      Broadcasts $\bar{I}_{ij}^h$ on communicator C_{ij}^{h-1}
    else
      Receive broadcast from communicator C_{ij}^{h-1}
    end if
    `TSQR` of columns $\bar{I}_{ij}^h$ of $\{A_{ij}, P_{ij} \in C_{ij}^h\} \rightarrow R$ factor on elected processor of C_{ij}^h
    Compute strong RRQR of $R \rightarrow I_{ij}^h$
  else
    if $\exists i', P_{i'j} \in \mathcal{C}_{ij}^h$ **then**
      Elected processor of $\mathcal{C}_{ij}^h \rightarrow P_{i'y}$
      Gather subcolumns I_{ij}^{h-1} of A_{ij} on $P_{iy}^h \rightarrow \bar{I}_{ij}^h$
      if $j = y$ ($P_{i'j}$ is elected on $\mathcal{C}_{ij}^h$) **then**
        `TSQR` of columns $\bar{I}_{ij}^h \rightarrow R$ factor on elected processor of C_{ij}^h
        Compute strong RRQR of $R \rightarrow I_{ij}^h$
      end if
    end if
  end if
end for

Ensure: I_{ij}^H indices of k columns of A

where

$$\begin{aligned} K_1 &= 16\log_2(P_r P_c) + 8\log_2(P_r)(1 + 5\log_2(P_r)/3) + \frac{16}{3}(5\log_2(P_r) - 1), \\ K_2 &= 8(\log_2(P_r) + 1). \end{aligned}$$

The number of messages is,

$$\#messages = (\log_2 P_c + \log_2 P_r)(1 + \log_2 P_r),$$

and the volume of communication is

$$\#words = (2k^2 + \frac{k}{2})(\log_2 P_r)(1 + \log_2 P_r) + 2k(\log_2 P_c)(m + 2k\log_2 P_r).$$

7 Parallel performance of QRTP

In this section we study the parallel performance of algorithm 3. We use a parallel machine formed by sixteen nodes composed of two Cascade Lake Intel Xeon 5218 with sixteen physical cores each. Each bi-processor has 192 GB shared memory. There are thus 1024 cores with 3,072 GB of distributed memory. The algorithm is implemented in C++, and built with GCC 8.3, OpenBLAS 0.3.7 and MPICH 3.3.1. For each run, the double precision matrix is generated using C++'s pseudo-random double precision number generator[1]. Then for P a given number of cores, the matrix is distributed over a $\sqrt{P} \times \sqrt{P}$ grid of processors and QRTP is applied.

[1] initialized with `std::uniform_real_distribution<double> dist(-32.768, 32.768)`, c.f. `https://en.cppreference.com/w/cpp/numeric/random/uniform_real_distribution`

Figure 11 shows the execution time of the algorithm for a 50 000 × 50 000 and a 100 000 × 100 000 randomly generated matrices, with the number of cores increasing from 16 to 1,024. Note that for the 100 000 × 100 000 matrix there is no runtime below 128 cores due to the limited shared memory available on each node. We observe that multiplying the number of processors by 8 from 128 to 1024 leads to a speedup of 5 for a 50 000 × 50 000 matrix and a speedup of 6 for a 100 000 × 100 000 matrix. Figure 12 presents weak scaling results when the dimensions of the submatrix per processor remain constant as 5000 × 5000. When the number of processors increase from 16 to 1024, the total dimensions of the matrix increase accordingly from 20 000 × 20 000 to 160 000 × 160 000. We observe that from 64 to 1024 cores the execution time is close to constant, which is a good result of weak scaling.

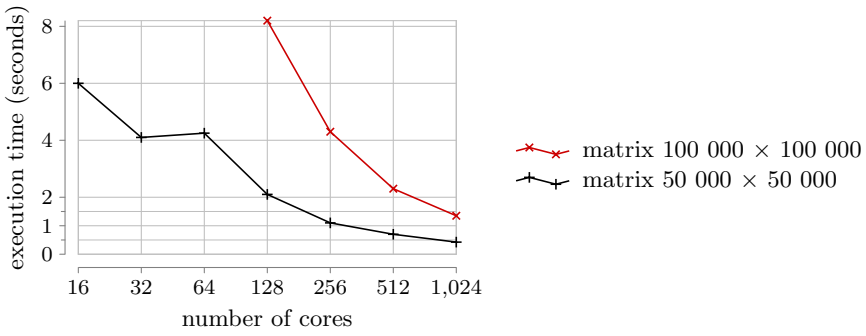

Figure 11: Strong scaling of algorithm 4 when selecting 10 columns. The dimensions of the random-generated matrix are fixed.

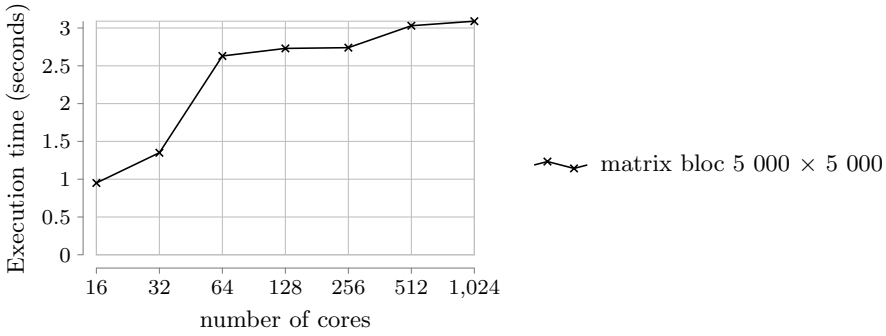

Figure 12: Weak scaling of algorithm 4 when selecting 10 columns. The dimensions of the random-generated matrix are fixed to 5 000 × 5 000 per core.

8 Conclusion

In this paper we introduced QRTP, a scalable communication avoiding algorithm using QR with tournament pivoting to compute a low rank approximation. QRTP guarantees bounds on the singular values of the resulting low rank approximation and provides accurate results in practice. QRTP for image compression show very close results to the sequential QRCP algorithm. We also presented an MPI implementation for QRTP performing well for strong and weak scaling on up to a thousand cores. Future work may address the possibility of using non-binary reduction trees and the associated trade-off between performance and accuracy, as well as the possibility of combining the selection of subsets of columns and of rows during the same tournament to obtain a CUR approximation.

9 Acknowledgements

This project has received funding from the European Research Council (ERC) under the European Unions Horizon2020 research and innovation program Grant agreement No. 810367.

References

[1] P. Businger and G. H. Golub. Linear least squares solutions by householder transformations. *Numerische Mathematik*, 7(3):269–276, June 1965.

[2] A. Carasso. Determining Surface Temperatures from Interior Observations. *SIAM Journal on Applied Mathematics*, 42(3):558–574, June 1982.

[3] T. F. Chan. Rank revealing QR factorizations. *Linear Algebra and its Applications*, 88-89:67–82, Apr. 1987.

[4] T. F. Chan and P. C. Hansen. Computing truncated singular value decomposition least squares solutions by rank revealing QR-factorizations. *Society for Industrial and Applied Mathematics. Journal on Scientific and Statistical Computing*, 11(3):519–530, 1990.

[5] J. Demmel, L. Grigori, M. Hoemmen, and J. Langou. Communication-optimal Parallel and Sequential QR and LU Factorizations. *SIAM Journal on Scientific Computing*, 34(1):A206–A239, Apr. 2008.

[6] J. Demmel, L. Grigori, and A. Rusciano. An improved analysis and unified perspective on deterministic and randomized low rank matrix approximations. *arXiv:1910.00223 [cs, math]*, Oct. 2019. arXiv: 1910.00223.

[7] J. W. Demmel, L. Grigori, M. Gu, and H. Xiang. Communication avoiding rank revealing QR factorization with column pivoting. *SIAM Journal on Matrix Analysis and Applications*, 36(1):55–89, 2015.

[8] Z. Drma and Z. Bujanovi. On the Failure of Rank-Revealing QR Factorization Software – A Case Study. *ACM Transactions on Mathematical Software*, 35(2):1–28, July 2008.

[9] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, Sept. 1936.

[10] G. H. Golub and C. F. Van Loan. *Matrix computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, fourth edition, 2013.

[11] M. Gu and S. C. Eisenstat. Efficient algorithms for computing a strong rank-revealing QR factorization. *SIAM*, 1996.

[12] Y. P. Hong and C.-T. Pan. Rank-revealing QR factorizations and the singular value decomposition. *Mathematics of Computation*, 58(197):213–213, Jan. 1992.

[13] C.-T. Pan and P. T. P. Tang. Bounds on singular values revealed by QR factorizations. *Bit Numerical Mathematics*, 39(4):740–756, 1999.

[14] G. Quintana-Ort, X. Sun, and C. H. Bischof. A BLAS-3 Version of the QR Factorization with Column Pivoting. *SIAM Journal on Scientific Computing*, 19(5):1486–1494, Sept. 1998.

[15] G. W. Stewart. Rank Degeneracy. *SIAM Journal on Scientific and Statistical Computing*, 5(2):403–413, June 1984.

[16] G. M. Wing and J. D. Zahrt. *A primer on integral equations of the first kind: the problem of deconvolution and unfolding*. Society for Industrial and Applied Mathematics, Philadelphia, 1991.